МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных технологий

Кафедра Информационных систем и технологий

Специальность 1-98 01 03 «Программное обеспечение информационной

безопасности мобильных систем»

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**К КУРСОВОМУ ПРОЕКТУ НА ТЕМУ:**

**Мобильное приложение железнодорожного вокзала**

Выполнил студент Тимошенко Дмитрий Валерьевич

(Ф.И.О.)

Руководитель проекта асс. Уласевич Н.И. (учен. степень, звание, должность, Ф.И.О., подпись)

И.о. зав. Кафедрой к.т.н., ст. преп. Блинова Е.А.

(учен. степень, звание, должность, Ф.И.О., подпись)

Консультант: асс. Уласевич Н.И. (учен. степень, звание, должность, Ф.И.О., подпись)

Нормоконтролер: асс. Уласевич Н.И. (учен. степень, звание, должность, Ф.И.О., подпись)

Курсовой проект защищен с оценкой

Минск 2024

**Содержание**

[Введение 5](#_Toc166794103)

[1. Аналитический обзор литературы по теме проекта 6](#_Toc166794104)

[1.1 Сравнительный анализ теоретических и эвристических методов решения поставленных задач и существующих по данной тематике технических научных решений 6](#_Toc166794105)

[1.2 Аналитический обзор аналогов 6](#_Toc166794106)

[1.2.1 Аналог «Белорусская железная дорога» 7](#_Toc166794107)

[1.2.2 Аналог «Intercity.pl» 8](#_Toc166794108)

[1.3 Изучение требований, определение вариантов использования 9](#_Toc166794109)

[1.3.1 Определение основных функциональных требований к программному средству 9](#_Toc166794110)

[1.3.2. Определение вариантов использования 10](#_Toc166794111)

[1.4. Вывод 10](#_Toc166794112)

[2. Проектирование приложения 11](#_Toc166794113)

[2.1 Обобщенная структура управления приложением 11](#_Toc166794114)

[2.2 Проектирование базы данных 13](#_Toc166794115)

[2.3 Описание информационых объектов и ограничений целостности 14](#_Toc166794116)

[2.3.1 Коллекция Stations 14](#_Toc166794117)

[2.3.2 Коллекция Schedule 14](#_Toc166794118)

[2.3.3 Коллекция Tickets 14](#_Toc166794119)

[2.3.4 Коллекция Trains 14](#_Toc166794120)

[2.3.5 Коллекция Users 15](#_Toc166794121)

[2.4 Вывод 15](#_Toc166794122)

[3. Реализация приложения 16](#_Toc166794123)

[3.1 Технические средства разработки 16](#_Toc166794124)

[3.2 Разработка мобильного приложения 16](#_Toc166794125)

[3.3 Вывод 18](#_Toc166794126)

[4. Тестирование приложения 19](#_Toc166794127)

[4.1 Вывод 21](#_Toc166794128)

[5. Руководство по использованию 22](#_Toc166794129)

[5.1 Работа с приложением 22](#_Toc166794130)

[5.1.1 Сторона пользователя 22](#_Toc166794131)

[5.1.2 Сторона менеджера 27](#_Toc166794132)

[5.1.3 Сторона администратора 27](#_Toc166794133)

[5.2 Вывод 28](#_Toc166794134)

[Заключение 29](#_Toc166794135)

[Список использованных литературных источников 30](#_Toc166794136)

[Приложение 31](#_Toc166794137)

[Приложение А 31](#_Toc166794138)

[Приложение Б 31](#_Toc166794139)

**Введение**

В современном мире разработка мобильных приложений играет важную роль в повышении эффективности и удобства использования различных сервисов. Одной из областей, где мобильные приложения могут быть особенно полезными, является железнодорожная отрасль. Разработка мобильного приложения для железнодорожного вокзала может значительно улучшить опыт путешествующих и обеспечить более эффективное управление информацией о состоянии вокзала и поездов.

Основная цель разработки такого приложения состоит в создании удобного и интуитивно понятного интерфейса для пользователей, позволяющего получать актуальную информацию о расписании поездов, изменениях маршрутов и других важных событиях на вокзале. Приложение должно предоставлять возможность быстрого и надежного поиска информации, а также уведомлять пользователей о любых изменениях.

Одной из важных задач разработки мобильного приложения для железнодорожного вокзала является интеграция с базой данных. Для эффективного хранения и обработки информации о поездах, расписании и других сущностях вокзала необходимо использовать базу данных. Одним из популярных вариантов баз данных является firebase.

В процессе разработки приложения необходимо провести аналитический обзор литературы по теме, изучить требования пользователей и определить варианты использования приложения. Затем следует провести анализ и проектирование модели данных, описать разработку мобильного приложения. Провести тестирование и написать руководство по использованию приложения.

Помимо этого, требуется описание используемых технологий разработки мобильного приложения.

В конечном итоге, курсовой проект должен быть завершен выводами по каждому разделу работы и заключением, включающим общий вывод по проделанной работе.

Разработка мобильного приложения для железнодорожного вокзала является актуальной и востребованной задачей, поскольку такое приложение может значительно улучшить взаимодействие пассажиров с железнодорожной инфраструктурой и обеспечить более комфортное и безопасное путешествие.

1. Аналитический обзор литературы по теме проекта

1.1 Сравнительный анализ теоретических и эвристических методов решения поставленных задач и существующих по данной тематике технических научных решений

Существует множество методов решения этой задачи, которые могут быть классифицированы по следующим параметрам:

1. Производительность: скорость выполнения запросов пользователя и эффективность использования ресурсов.
2. Надежность: способность системы работать без сбоев, а также ее способность восстановиться после сбоя.
3. Гибкость: способность системы адаптироваться к изменениям требований и удовлетворять различным потребностям пользователей.
4. Расширяемость: возможность системы расширяться и добавлять новые функциональные возможности.

В области мобильной разработки применяются различные технические решения, основанные на принципах ООП. Например, фреймворки, такие как React Native и Flutter, предоставляют инструменты для создания кроссплатформенных приложений, поддерживающих принципы инверсии управления и управления состоянием данных.

Теоретические подходы в мобильной разработке включают формальные методы, такие как моделирование пользовательского интерфейса, анализ потребностей пользователей и применение принципов SOLID для проектирования устойчивых и модульных систем.

Эвристические методы в мобильной разработке основываются на опыте и лучших практиках, таких как использование шаблонов проектирования для решения стандартных задач интерфейса и архитектуры. Книги, такие как “Design Patterns: Elements of Reusable Object-Oriented Software”[1], предлагают ценные ресурсы для разработчиков, стремящихся создать гибкие и масштабируемые мобильные приложения.

Кроме того, важным аспектом мобильной разработки является управление жизненным циклом приложения. Это включает в себя оптимизацию поведения приложения при различных состояниях устройства, таких как режим ожидания, активное использование или переход между страницами. Разработчики используют различные методы для обеспечения плавности работы приложения, минимизации потребления энергии и сохранения данных пользователя. Например, паттерны MVC (Model-View-Controller) и MVVM (Model-View-ViewModel) помогают в структурировании кода и управлении состоянием, что критически важно для создания отзывчивых и надежных мобильных приложений.

1.2 Аналитический обзор аналогов

В современном мире, когда скорость и точность играют важную роль, транспортные компании становятся все более популярными. Одной из главных задач таких компаний является надежная доставка клиентов в места назначения.

1.2.1 Аналог «Белорусская железная дорога»

В качестве аналога рассмотрим приложение белорусской железной дороги [2], которое представлено на рисунке 1.1.

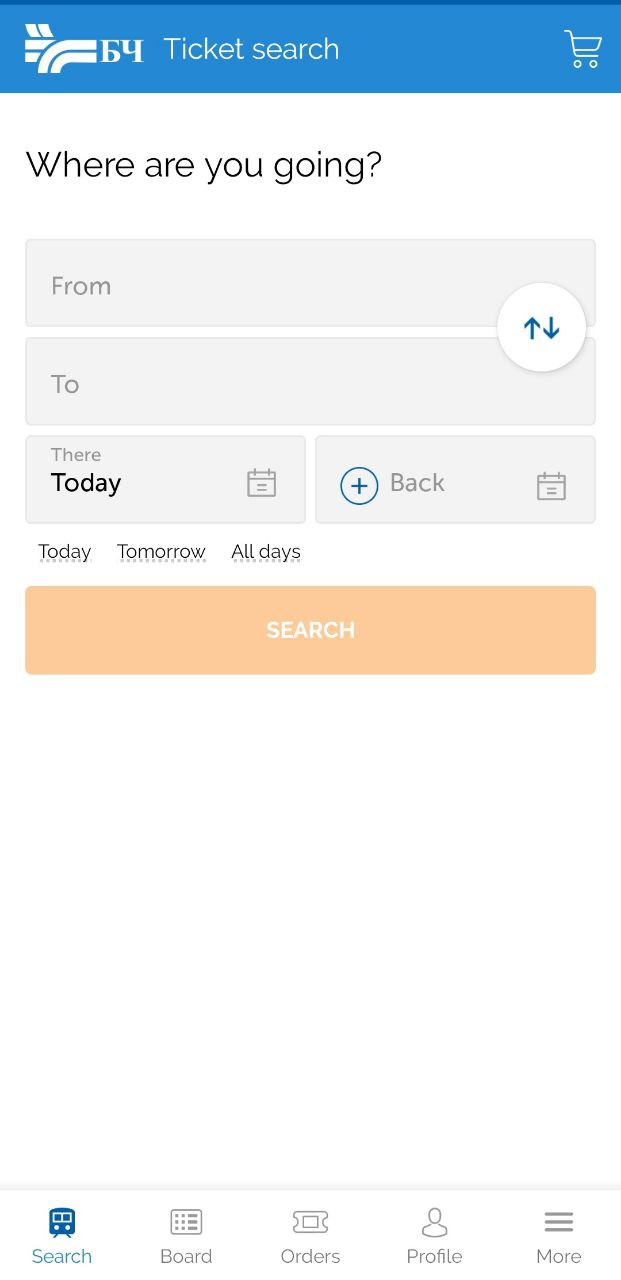


Рисунок 1.1 – Приложение «БЧ. Мой поезд»

Данное приложение позволяет из любой точки страны, где имеется доступ в интернет, забронировать билет на поезд. Пример представлен на рисунке 1.2.

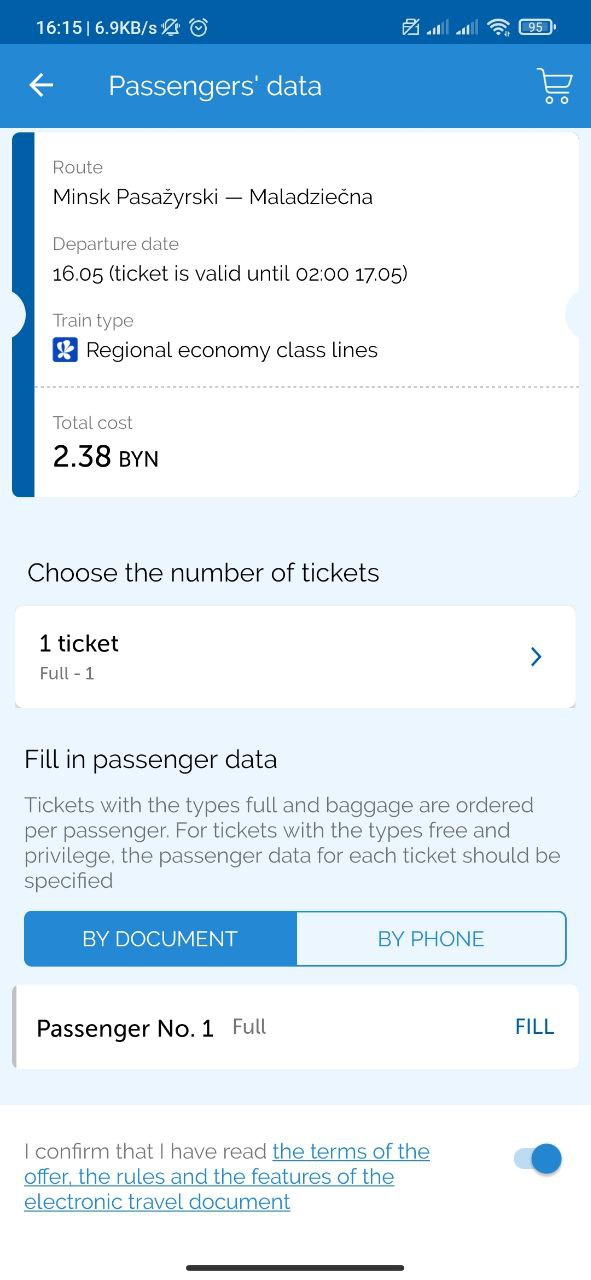


Рисунок 1.2 – Бронирование билета в приложении «БЧ. Мой поезд»

Также приложение позволяет просмотреть расписание рейсов онлайн. Пример на рисунке 1.3.

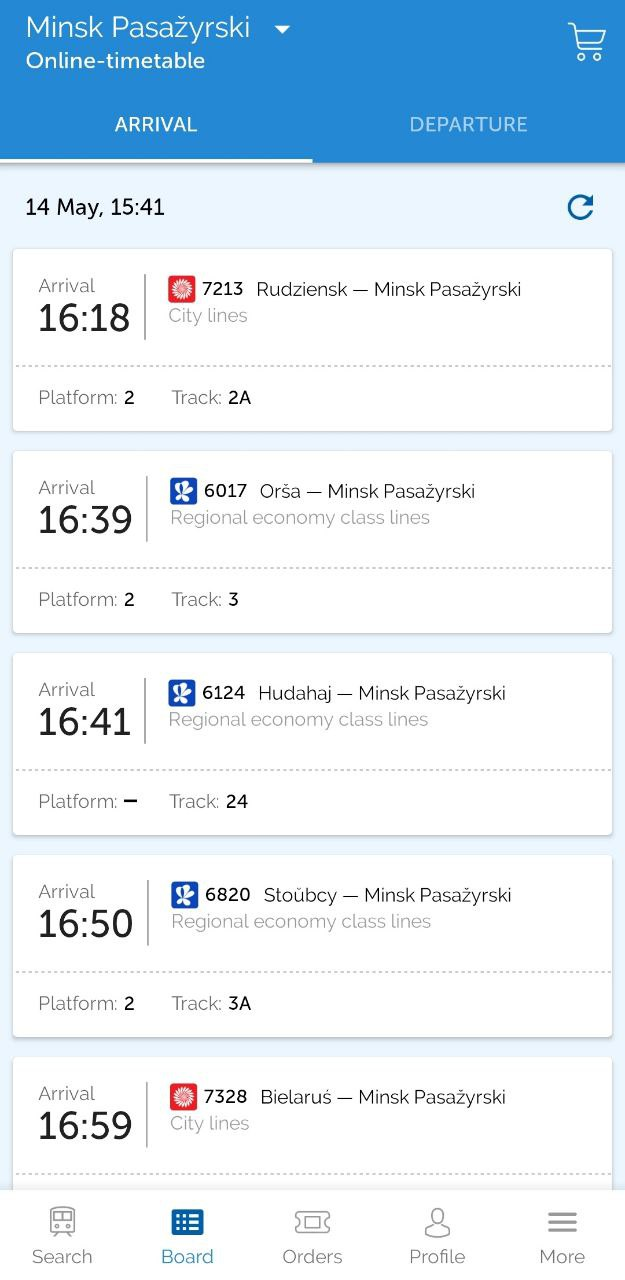


Рисунок 1.3 – Расписание рейсов в приложении «БЧ. Мой поезд»

Однако, как и любое подобное приложение, у него есть минусы. Одним из таких минусов можно назвать ошибки при бронировании билетов. У некоторых пользователей возникают проблемы с бронированием билетов через приложение. Например, система может выдавать ошибки при оплате или не отображать доступные для бронирования билеты.

1.2.2 Аналог «Intercity.pl»

Еще одним аналогом является сайт «Intercity.pl» [3], предоставляющий услуги по бронированию билетов на территории Польши. Сайт продемонстрирован на рисунке 1.4.

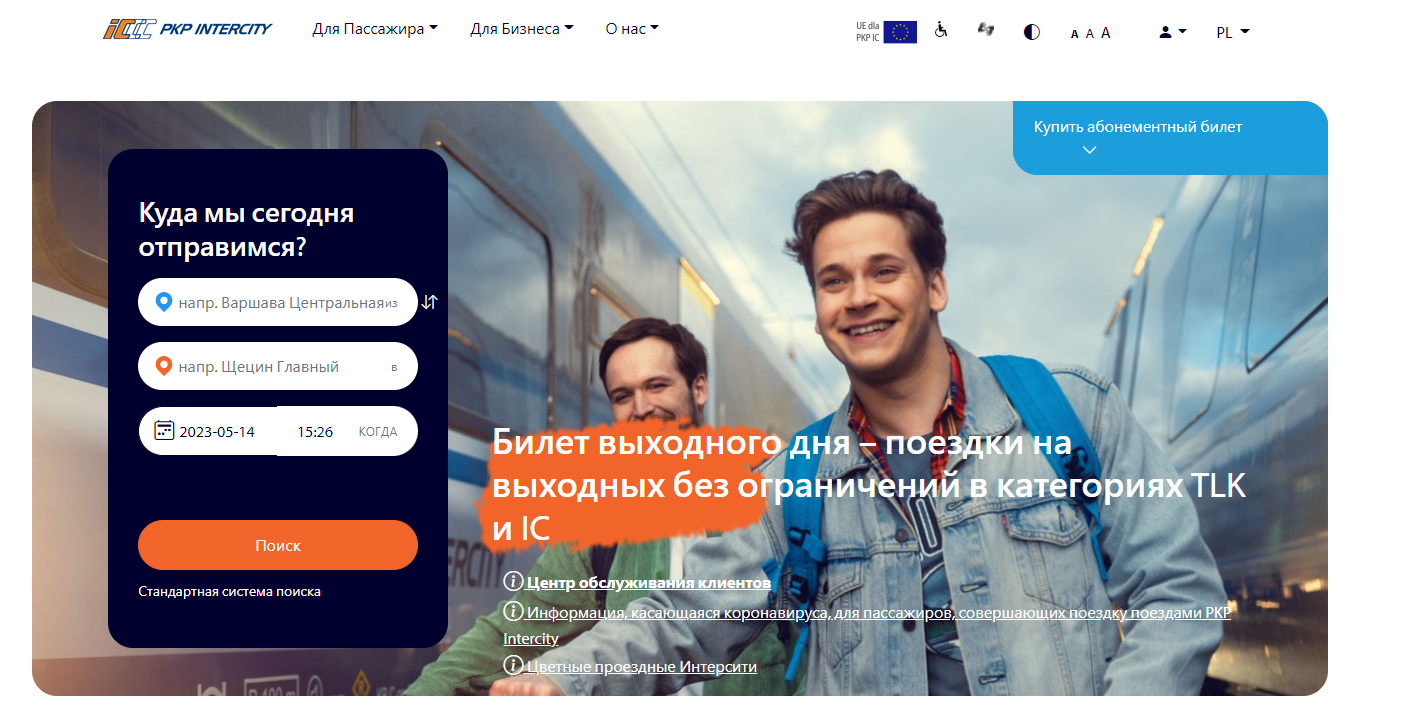


Рисунок 1.4 – Сайт «Intercity.pl»

Из минусов, можно отметить, наличие только возможности поиска билетов и бронирования билетов.

1.3 Изучение требований, определение вариантов использования

1.3.1 Определение основных функциональных требований к программному средству

Различают три уровня требований к проекту:

* бизнес-требования;
* пользовательские требования;
* функциональные требования.

Бизнес-требования содержат высокоуровневые цели организации или заказчиков системы. Курсовой проект не подразумевает наличие заказчика, который мог бы выдвинуть бизнес-требования, поэтому в качестве таких высокоуровневых требований можно рассматривать общие требования к разрабатываемому средству. К их числу относятся:

* простота и лёгкость интерфейса;
* использование базы данных;

Весь дальнейший процесс проектирования и разработки программного средства должен находиться в очерченных бизнес-требованиями границах.

Следующими требованиями являются требования пользователей. Данные требования описывают цели и задачи, которые пользователям позволит решить система. Таким образом, в пользовательских требованиях указано, что клиенты смогут делать с помощью системы. Пользователь данного программного решения должен иметь возможность:

* входить в приложение, после ввода необходимых данных;
* просматривать доску расписания и свои забронированные билеты;
* выполнять поиск необходимых рейсов;
* бронировать билеты;
* изменять свои персональные данные.

Менеджер имеет возможность:

* Добавлять билеты;
* Изменять и удалять брони билетов;
* Редактировать данные о пользователях.

Администратор имеет возможность:

* Поддерживать работу с базой данных;
* Изменять данных, хранящиеся в базе данных.

После проведения анализа были выявлены следующие функциональные требования:

* вся информация должна храниться в базе данных;
* приложение должно производить валидацию вводимых пользователем данных;
* приложение должно корректным образом обрабатывать возникающие исключительные ситуации: отображать понятное для пользователя сообщение о возникшей ошибке;
* приложение должно предоставлять возможность пользователям проходить аутентификацию и входить в систему под соответствующим введенным данным пользовательским именем;
* приложение должно предоставлять возможность получения информации по различным критериям;
* приложение должно предоставлять информацию о расписании движения поездов пассажирам.
* приложение должно позволять создавать, изменять, удалять данных из базы данных.
* приложение должно отображать информацию о билетах на поезда, включая маршрут, дату и время отправления, место в вагоне.
* приложение должно позволять выполнять поисковые запросы и фильтрацию.
* приложение должно предоставлять интерфейс для бронирования билетов.
* приложение должно обеспечивать защиту данных и доступ к базе данных.

Таким образом, был проведен тщательный анализ требований к программному средству, который позволил разработать список функциональных требований. Разработка данной программной системы должна проводиться в соответствии с сформированными списком.

1.3.2. Определение вариантов использования

Вариантами использования программного средства являются:

1. Авторизация пользователей, менеджеров или администраторов.
2. Просмотр расписания поездов.
3. Просмотр информации о наличии мест на поездах.
4. Бронирование места на поезд.
5. Создание расписания со стороны администратора.
6. Просмотр оформленных билетов.
7. Администрирование, назначенным администратором.

1.4. Вывод

В данном разделе проведен аналитический обзор литературы по теме проекта, а также осуществлен сравнительный анализ теоретических и эвристических методов решения поставленных задач и существующих технических научных решений.

Также был проведен аналитический обзор аналогов, включающий два конкретных примера аналогов: «Белорусская железная дорога» и «Intercity.pl». Это позволило определить преимущества и недостатки существующих решений и использовать эту информацию при разработке нового проекта.

2. Проектирование приложения

2.1 Обобщенная структура управления приложением

Пользовательский интерфейс взаимодействует с бизнес-логикой, которая в свою очередь взаимодействует со слоем доступа к данным. Слой доступа к данным выполняет запросы к базе данных и возвращает результаты обратно в бизнес-логику, которая обрабатывает эти результаты и возвращает их обратно в пользовательский интерфейс. База данных содержит данные, с которыми приложение работает.

Обобщенная структура управления приложением представлена на рисунке 2.1.

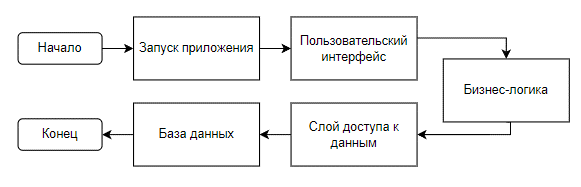


Рисунок 2.1 – Обобщенная структура управления приложением

Пользовательский интерфейс играет ключевую роль в успешном мобильном приложении, поскольку он формирует первое впечатление и определяет взаимодействие пользователя с приложением. В рамках данного проекта, ориентированного на мобильные платформы, важно уделить особое внимание созданию высококачественного и интуитивно понятного пользовательского интерфейса.

Для организации клиентской части приложения будет использован модульный подход, разделяющий код на две основные области: взаимодействие с пользовательским интерфейсом и внутренние процессы приложения. Это позволит работать над каждой частью независимо друг от друга, повышая гибкость и масштабируемость разработки.

Навигация в мобильном приложении является критически важной, обеспечивая логическую связь между различными экранами и страницами. При проектировании навигации будет уделено особое внимание интуитивности и удобству перемещения пользователя внутри приложения.

При первом запуске приложения пользователь попадает на страницу поиска, где он может найти нужный маршрут, посмотреть наличие билетов на поезд и оформить их при наличии, либо воспользоваться панелью навигацией, прикрепленной к нижней части экрана.

Панель навигации позволяет переместиться на страницы: просмотра расписания, просмотра оформленных заказов, входа, регистрации, профиля и страницы информации о приложении. Схема навигации для пользователя представлена на рисунке 2.2.

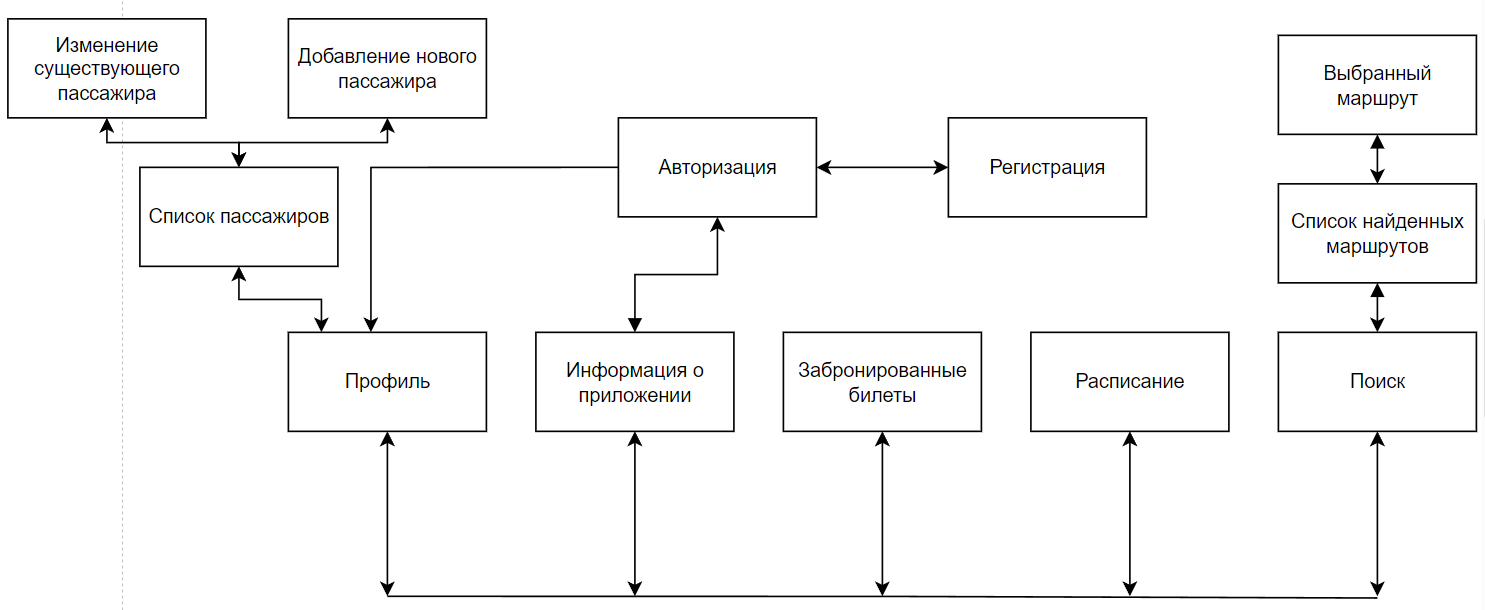


Рисунок 2.2 – Схема навигации для пользователя

Если пользователь решил работать с приложением без входа в систему, то у него доступны только базовые страницы: поиска и просмотра доступных билетов, расписания, информации о приложении и страницы входа и регистрации. Схема продемонстрирована на рисунке 2.3.

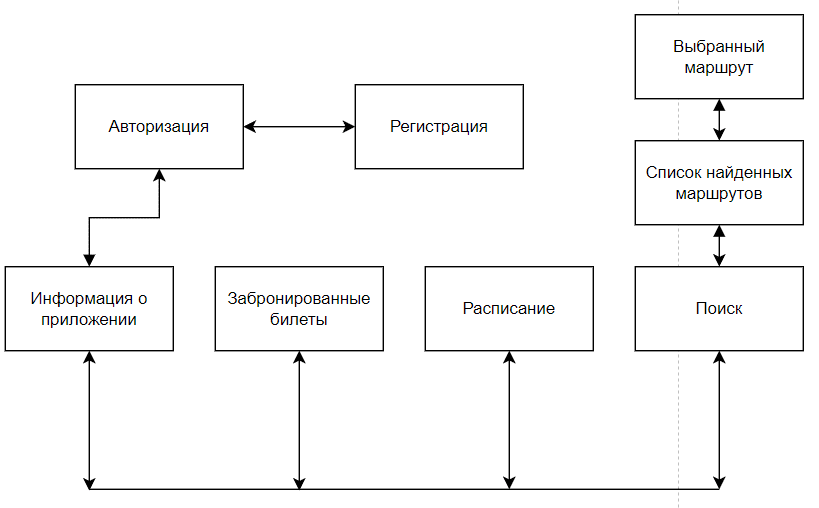


Рисунок 2.3 – Схема навигации для неавторизованного пользователя

Для администратора и менеджера после авторизации доступна дополнительная страница: Панель администратора.

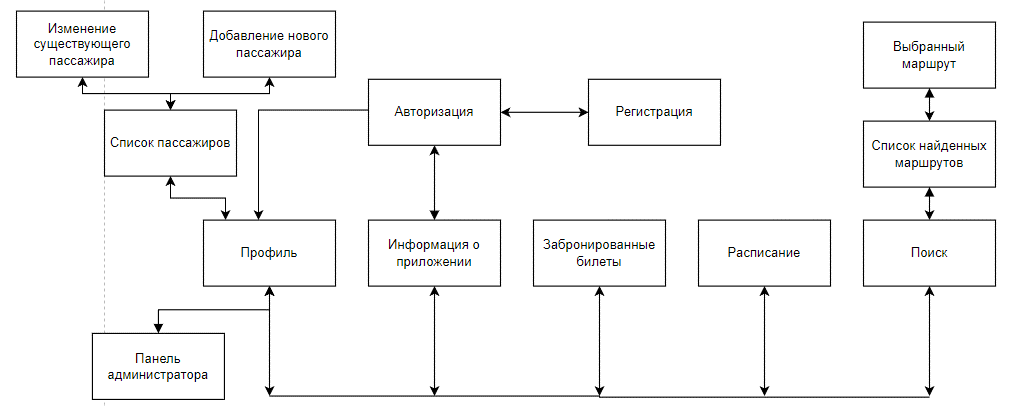


Рисунок 2.4 – Схема навигации администратора

На панели администратора админ может создавать, изменять и удалять данные из всех коллекций, а менеджер только у коллекций users и tickets.

Диаграмма вариантов использований представлена в приложении А.

2.2 Проектирование базы данных

Для разработки веб-приложения в рамках курсового проекта понадобилась база данных с необходимой конфигурацией сущностей для хранения всей необходимой информации. Структуру базу данных, ограничения целостности, связи и поля можно увидеть в приложении Б и на рисунке 2.5.

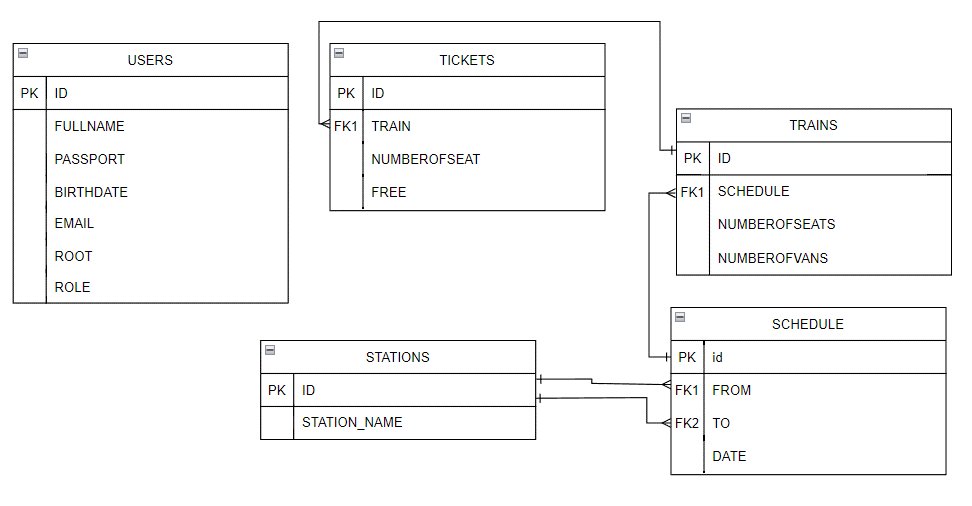


Рисунок 2.5 – Структура базы данных

Созданная база данных содержит в себе 5 коллекций: Users, Schedule, Tickets, Trains и Stations.

2.3 Описание информационых объектов и ограничений целостности

Далее будет описана структура каждой коллекции. В коллекции users хранятся данные для идентификации пользователей, зарегистрированных в приложении, schedule – хранение данных о расписании, tickets для хранения существующих билетов, trains для хранения данных о поездах (составах) и stations для хранения названий станций.

2.3.1 Коллекция Stations

В её состав входят следущие столбцы:

* Поле id. Хранит уникальный идентификатор каждой станции.
* Поле name. Хранит название станции.

Данная коллекция содержит ограничение первичного ключа для поля id, а также все поля должны содержать какое-либо значение.

2.3.2 Коллекция Schedule

В её состав входят следущие столбцы:

* Поле id. Хранит уникальный идентификатор каждой ячейки расписания.
* Поле from. Хранит ID начальной точки.
* Поле to. Хранит ID конечной точки.
* Поле date. Хранит дату и время отправления.

Данная коллекция содержит ограничение первичного ключа для поля id, все поля должны содержать какое-либо значение.

Также для полей from и to существуют ограничения целостности по внешнему ключу к коллекции stations.

2.3.3 Коллекция Tickets

В её состав входят следущие столбцы:

* Поле id. Хранит уникальный идентификатор каждого билета.
* Поле train. Хранит ID поезда, к которому привязан данный билет.
* Поле numberOfSeat. Хранит номер забронированного места в поезде.
* Поле free. Хранит строку «true», либо ID пользователя, забронировавшего билет.

Данная коллекция содержит ограничение первичного ключа для поля id, все поля должны содержать какое-либо значение.

Также для поля train существует ограничение целостности по внешнему ключу к коллекции trains.

Также коллекция включает ограничение check на поле free.

2.3.4 Коллекция Trains

В её состав входят следущие столбцы:

* Поле id. Хранит уникальный идентификатор каждого состава.
* Поле schedule. Хранит ID ячейки расписания, к которому состав привязан.
* Поле numberOfSeats. Хранит число, указывающее на количество мест в вагоне.
* Поле numberOfVans. Хранит число, обозначающее количество вагонов в составе.

Данная коллекция содержит ограничение первичного ключа для поля id, все поля должны содержать какое-либо значение.

2.3.5 Коллекция Users

В её состав входят следущие столбцы:

* Поле id. Хранит уникальный идентификатор каждого пользователя.
* Поле fullName. Хранит ФИО пользователя.
* Поле passport. Хранит номер паспорта пользователя.
* Поле birthdate. Хранит дату рождения пользователя.
* Поле email. Хранит адрес электронной почты пользователя.
* Поле root. Хранит ID пользователя, для организации иерархии.
* Поле role. Хранит роль пользователя.

Данная коллекция содержит ограничение первичного ключа для поля id.

Также коллекция включает ограничение check на столбец role.

2.4 Вывод

В данном разделе была разработана архитектура проекта, описано взаимодействие приложения с базой данных, составлена диаграмма вариантов использования. Также была описана структура базы данных вместе с ограничениями целостности.

3. Реализация приложения

Реализация программного средства — это процесс преобразования детального плана или концепции программного продукта, разработанного в процессе проектирования, в функциональный продукт.

3.1 Технические средства разработки

Для реализации курсового проекта будет использован язык программирования kotlin, а также фреймворк Jetpack Compose.

Kotlin — это современный, статически типизированный язык программирования, который используется в качестве основного языка разработки для Android-приложений. Он обладает высокой производительностью, лаконичностью и безопасностью кода.

Jetpack Compose — это современная библиотека UI для Android, разработанная Google, которая позволяет создавать пользовательские интерфейсы с помощью декларативного подхода. Она обеспечивает высокую производительность и гибкость при разработке интерфейсов.

Dagger Hilt — это библиотека для внедрения зависимостей в Android-приложения, основанная на популярной библиотеке Dagger. Она упрощает управление зависимостями и повышает модульность кода.

Firebase — это облачная платформа разработки приложений, предоставляемая компанией Google. Она включает в себя следующие ключевые компоненты:

Firebase Authentication: Сервис аутентификации, который обеспечивает безопасную регистрацию и аутентификацию пользователей в приложении.

Cloud Firestore: Гибкая и масштабируемая база данных в реальном времени для хранения и синхронизации данных.

Firebase Storage: Сервис хранения файлов, который позволяет загружать и скачивать файлы, такие как изображения, видео и документы.

Material UI — это библиотека компонентов и стилей, основанная на дизайн-системе Material Design от Google. Она обеспечивает готовые UI-элементы, которые соответствуют современным стандартам дизайна и повышают визуальную привлекательность приложения.

Credential Manager — это библиотека, которая помогает управлять учетными данными пользователей, такими как логины и пароли, в Android-приложениях. Она обеспечивает безопасное хранение и использование этих данных.

Таким образом, данное приложение будет построено с использованием следующего стека технологий: Kotlin, Jetpack Compose, Dagger Hilt, Firebase (Authentication, Cloud Firestore, Storage), Material UI и Credential Manager, а также других вспомогательных библиотек.

3.2 Разработка мобильного приложения

Мобильное приложение состоит из набора паттернов и пакетов. Каждый пакет собирает в себе похожие по назначению классы:

* Models – представляет модели, которые используются для взаимодействия с базой данных. Здесь находятся классы моделей данных, которые отображают структуру данных приложения.
* Services – содержит классы-сервисы, использующиеся во всем приложении.
* Ui.theme – содержит классы, отвечающие за настройку доступных тем, цветовой палитры и шрифтов приложения, а также пользовательские UI-компоненты.
* ViewModels – содержит классы, содержащие логику взаимодействия с UI.
* Views – содержит классы, предоставляющие пользовательский интерфейс.

При разработке были использованы паттерны проектирования, такие как: MVVM, Singleton, Repository, State, Factory Method.

Реализация асихронной функции, которая выполняет авторизацию пользователя через Google аккаунт, представлена в листинге 3.1.

suspend fun signUpWithCredentials(

context: Context,

credentialManager: CredentialManager,

request: GetCredentialRequest,

onCompleteListener: (DocumentSnapshot) -> Unit,

onFailureListener: (Exception) -> Unit

) {

try {

val result = credentialManager.getCredential(

request = request,

context = context

)

val credential = result.credential

val googleIdToken = GoogleIdTokenCredential.createFrom(credential.data).idToken

val firebaseCredentials = GoogleAuthProvider.getCredential(googleIdToken, null)

val resultAuth = auth.signInWithCredential(firebaseCredentials).await()

val user = User(

id = resultAuth.user!!.uid,

fullName = resultAuth.user!!.displayName.toString(),

email = resultAuth.user!!.email.toString(),

role = "user",

root = resultAuth.user!!.uid

)

var doc = db.collection("users").document(user.id).get().await()

if (!doc.exists()) {

db.collection("users").document(user.id).set(user.toFirebase()).await()

doc = db.collection("users").document(user.id).get().await()

}

onCompleteListener(doc)

} catch (e: Exception) {

onFailureListener(e)

Log.e("Error", "signInGoogle")

Log.e("Error", e.message.toString())

}

}

Листинг 3.1 – Функция авторизации через Google

Она принимает credentialManager, для хранения данных авторизации, GetCredentialRequest, для получения запроса на получения данных об Google аккаунте, и колбеки, для действий после выполнения функции и работы, если произошла ошибка.

**3.3 Вывод**

В данном разделе были рассмотрены аспекты разработки мобильного приложения. Были описаны технические средства, используемые для разработки.

4. Тестирование приложения

Тестирование приложения является неотъемлемой частью процесса разработки, позволяющей обеспечить высокое качество продукта и удовлетворенность пользователей.

Для начала протестируем страницу авторизации. Если пользователь ввел неверные данные или наоборот оставил поля пустыми, то будет отображено соответсвующее сообщение, показанное на рисунке 4.1.

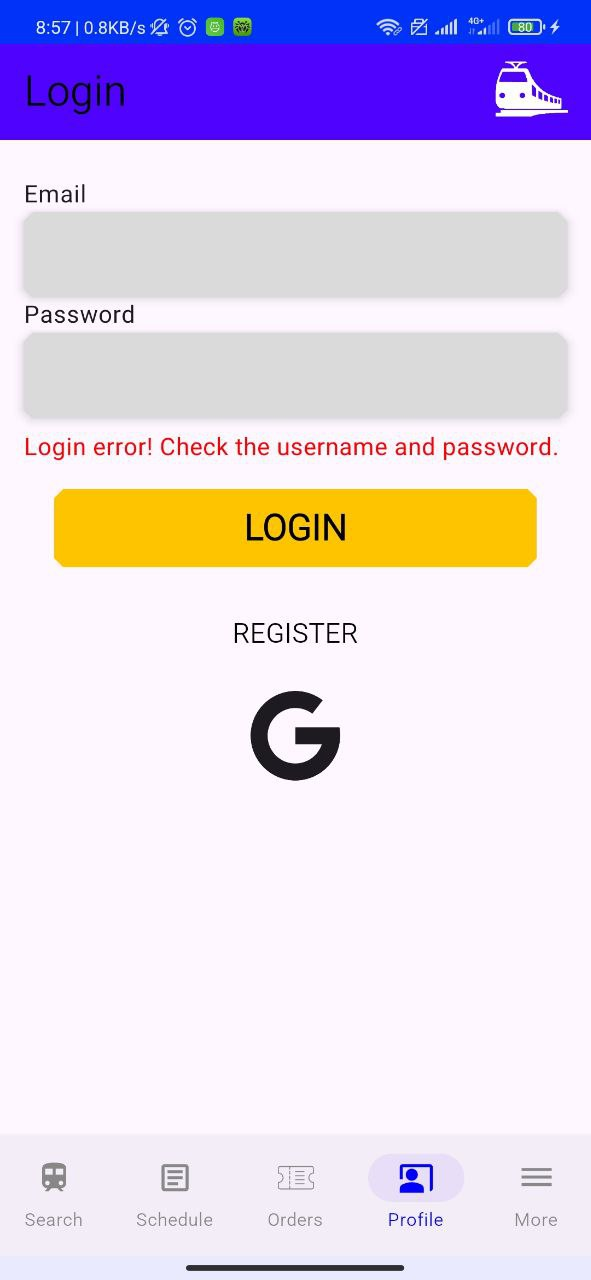


Рисунок 4.1 – Тестирование авторизации

Далее протестируем страницу регистрации. Пользователь может зарегистрироваться с уникальным email, ему необходимо заполнить все поля, также ему необходимо повторить введённый пароль. Кроме того, есть проверка на корректность всех полей. Тестирование продемонстрировано на рисунке 4.2.

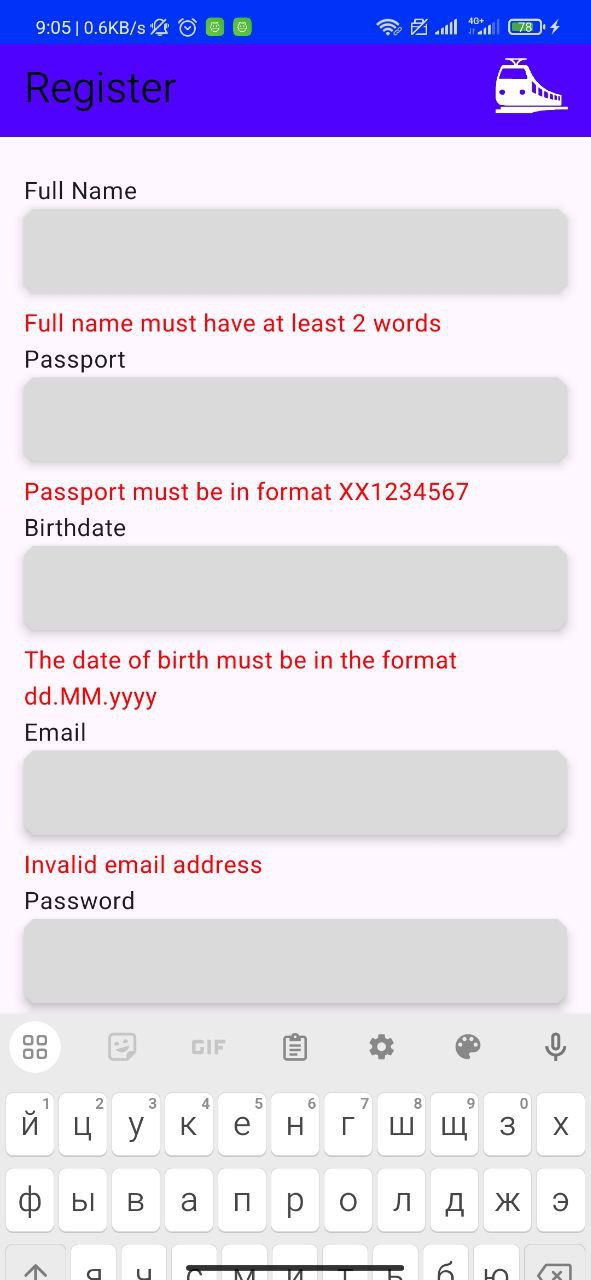


Рисунок 4.2 – Тестирование регистрации

После авторизации или регистрации пользователь попадает на страницу профиля. Протестируем добавление нового пассажира. При добавлении нужно нажать на кнопку «Пассажиры», нажать на кнопку «Добавить пассажира» ввести данные о пассажире. Их валидация совпадает с валидацией данных при регистрации. Тестирование отображено на рисунке 4.3.



Рисунок 4.3 – Тестирование добавления пассажира

Одной из важных функций приложения является поиск билетов, поэтому существует необходимость описать их тестирование. На странице поиска при выборе промежутка интересующих дат, реализована проверка промежутка, если он введен неверно, то дата возвращения будет игнорироваться. Тестирование отображено на рисунке 4.4.

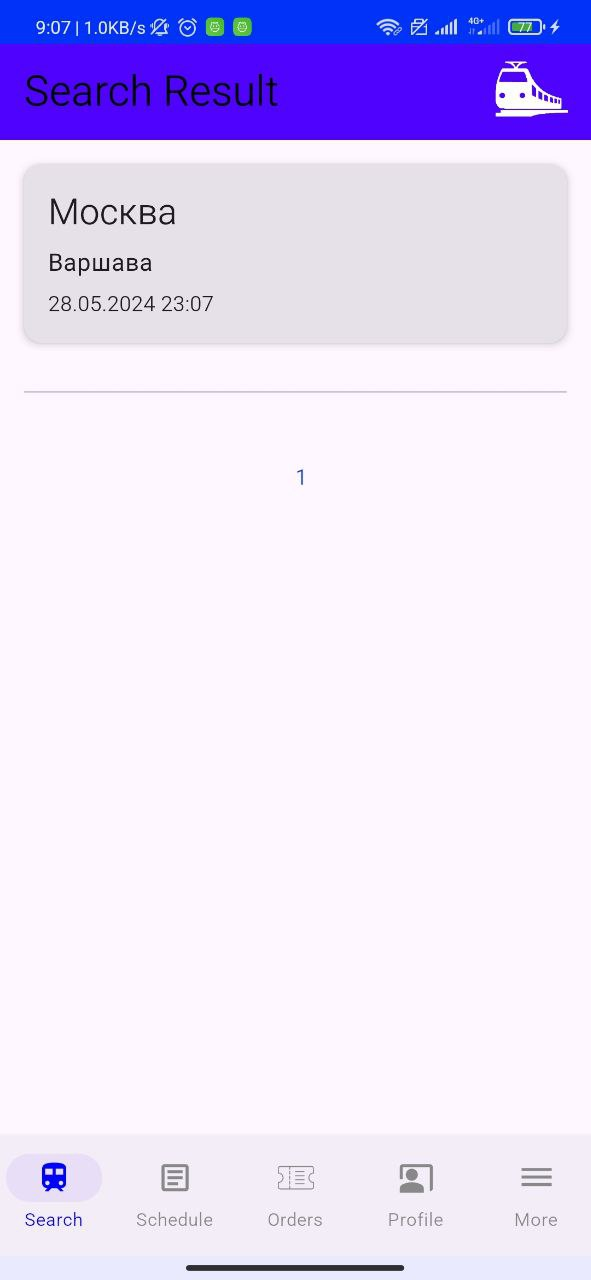
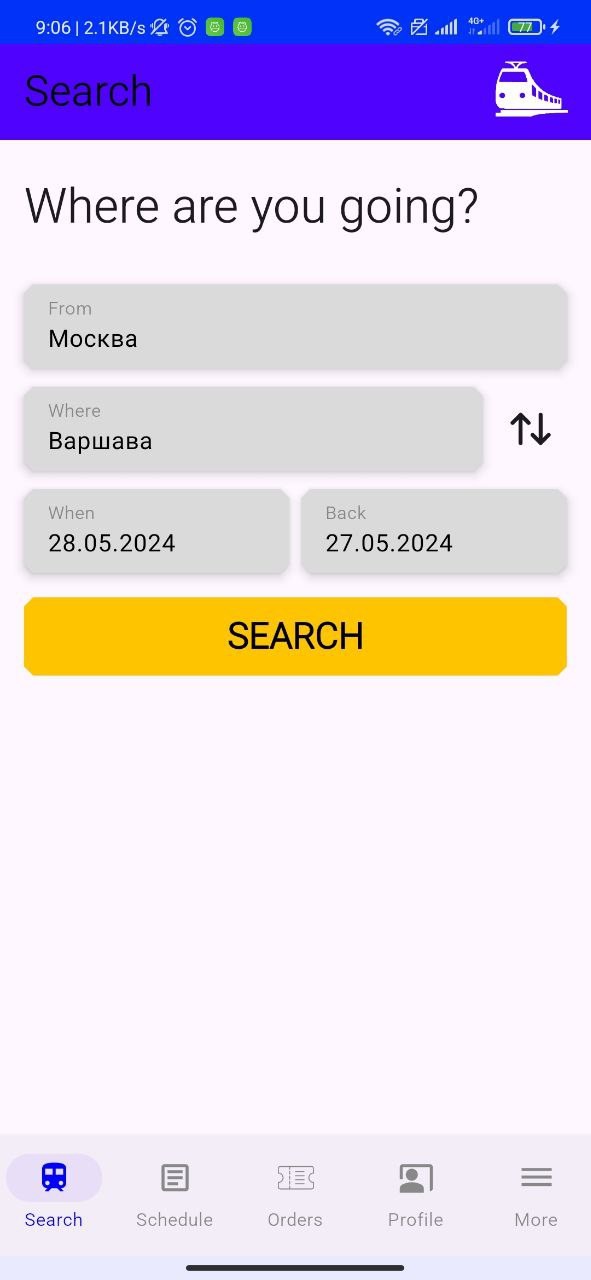


Рисунок 4.4 – Тестирование выбора промежутка дат поездок

Также в приложении возможно бронирование билетов. На странице выбора места для бронирования, при клике на место всплывает окошко, где необходимо выбрать пассажира для оформления билета. Если пользователь не хочет выбирать пользователя, то кнопка подтверждения будет неактивной и оформить билет не представляется возможным. Тестирование отображено на рисунке 4.5.

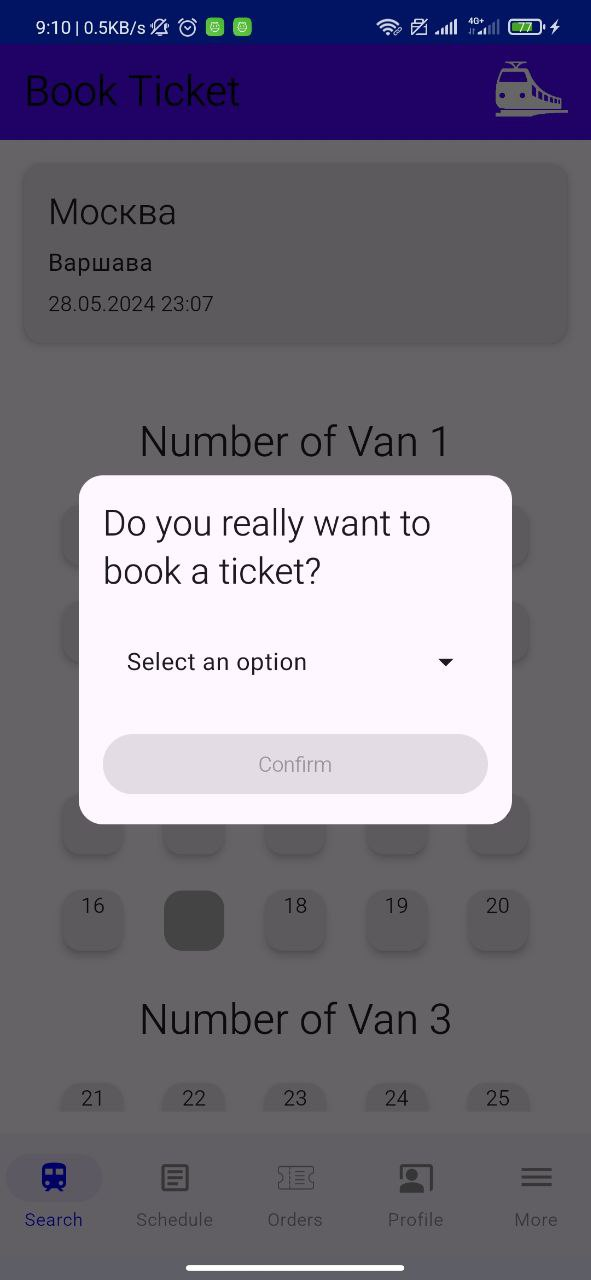


Рисунок 4.5 – Тестирование бронирования билета

Кроме рассмотренных возможностей приложения, обработаны возможные ошибки при редактировании существующего пассажира.

4.1 Вывод

В данном разделе были протестированы основные функции приложения. По итогам тестирования можно сказать, что все работает корректно.

5. Руководство по использованию

5.1 Работа с приложением

5.1.1 Сторона пользователя

Приложение поддерживает русский и английский языки.

У пользователей есть возможность пройти авторизацию либо регистрацию. Для регистрации ему требуется ввести ФИО, свой email, паспорт, дату рождения, а также придумать пароль для учетной записи. Страницы регистрации и авторизации показаны на рисунке 5.1.

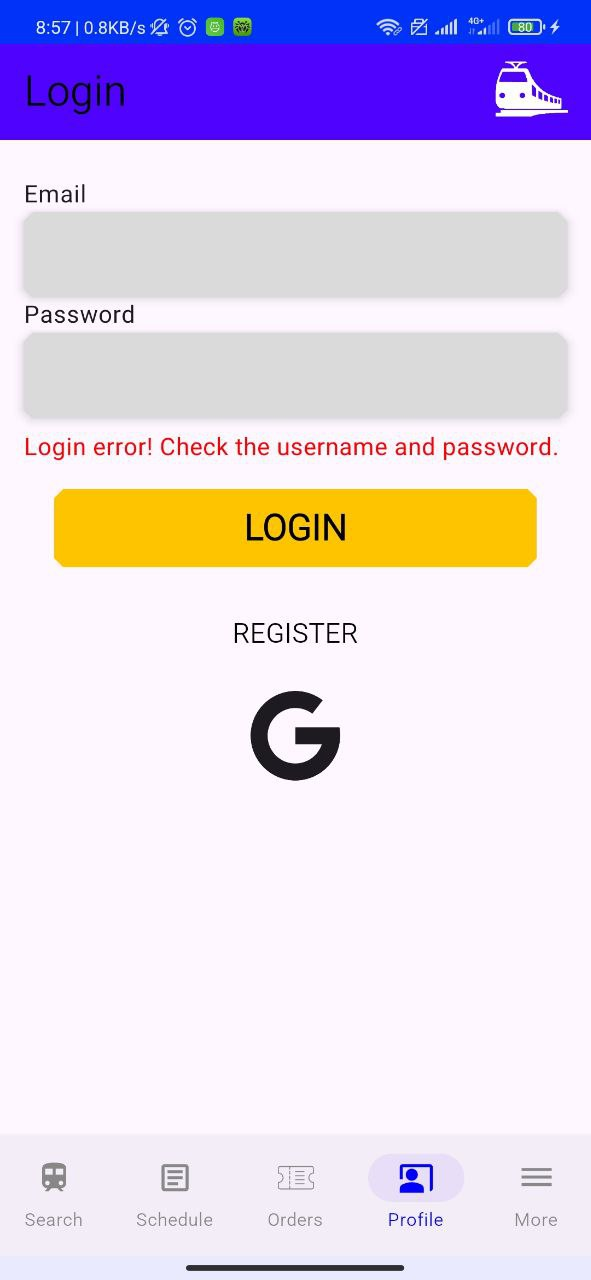
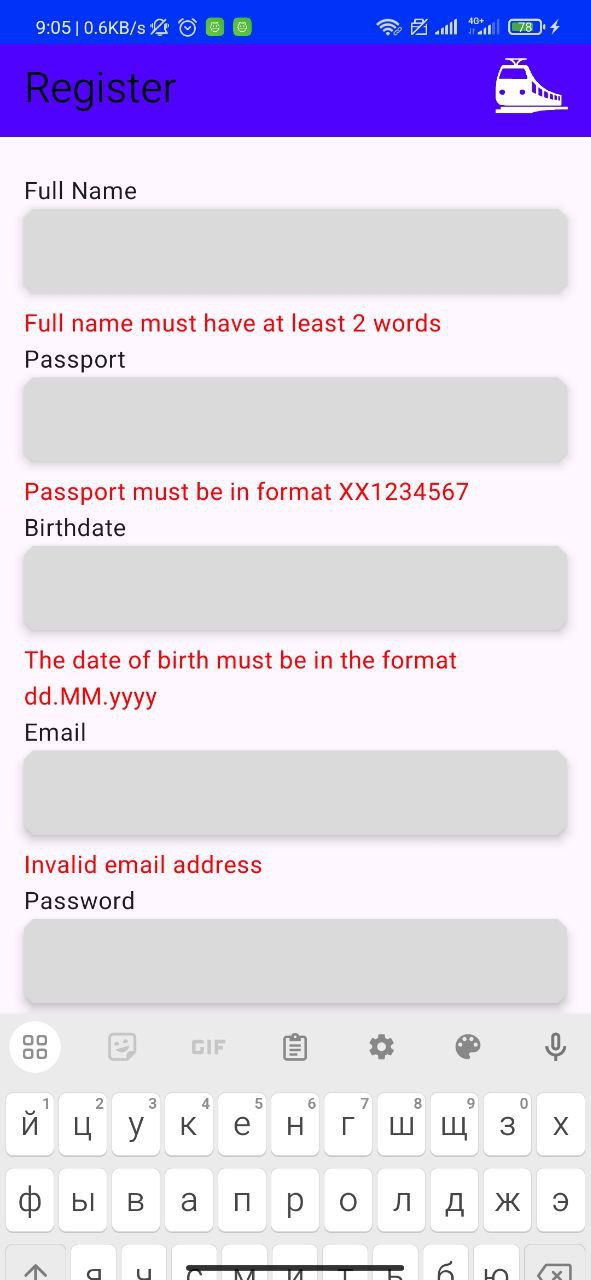


Рисунок 5.1 – Страницы Регистрации и Авторизации

При прохождении авторизации, откроется страница профиля, состоящей из списка кнопок. Страница профиля отображена на рисунке 5.2.

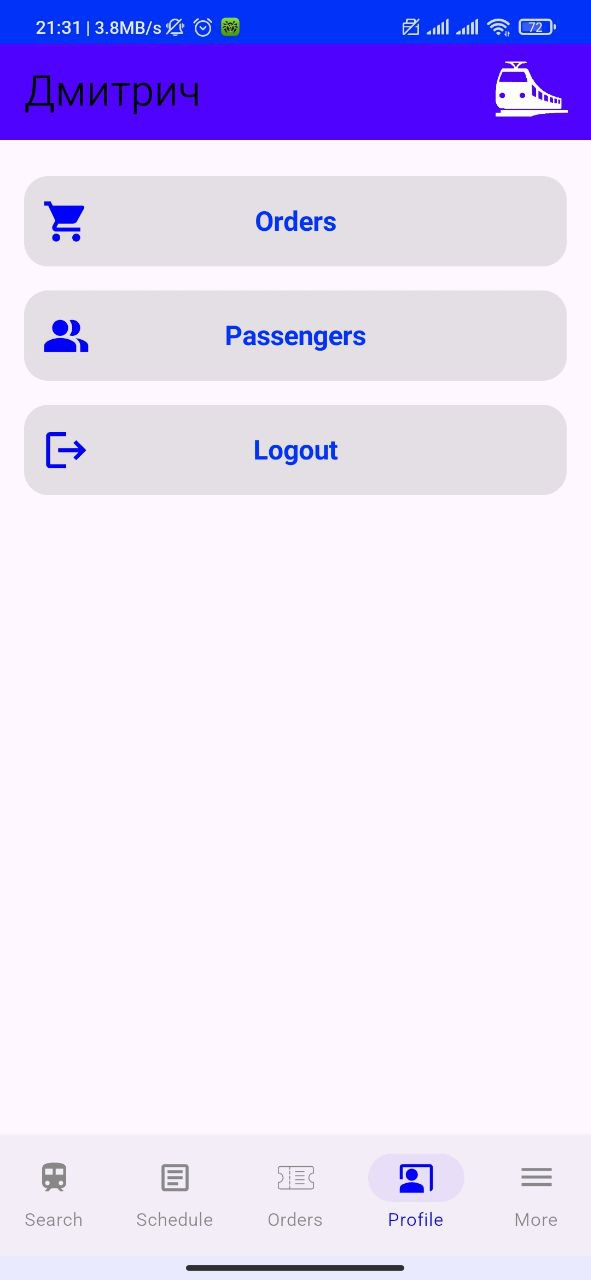


Рисунок 5.2 – Страницы просмотра списка пассажиров и редактирование существующего пассажира

Одна из возможностей авторизованного пользователя – это добавление или редактирование существующего пассажира. Страницы просмотра списка пассажиров и редактирование существующего продемонстрированы на рисунке 5.3.

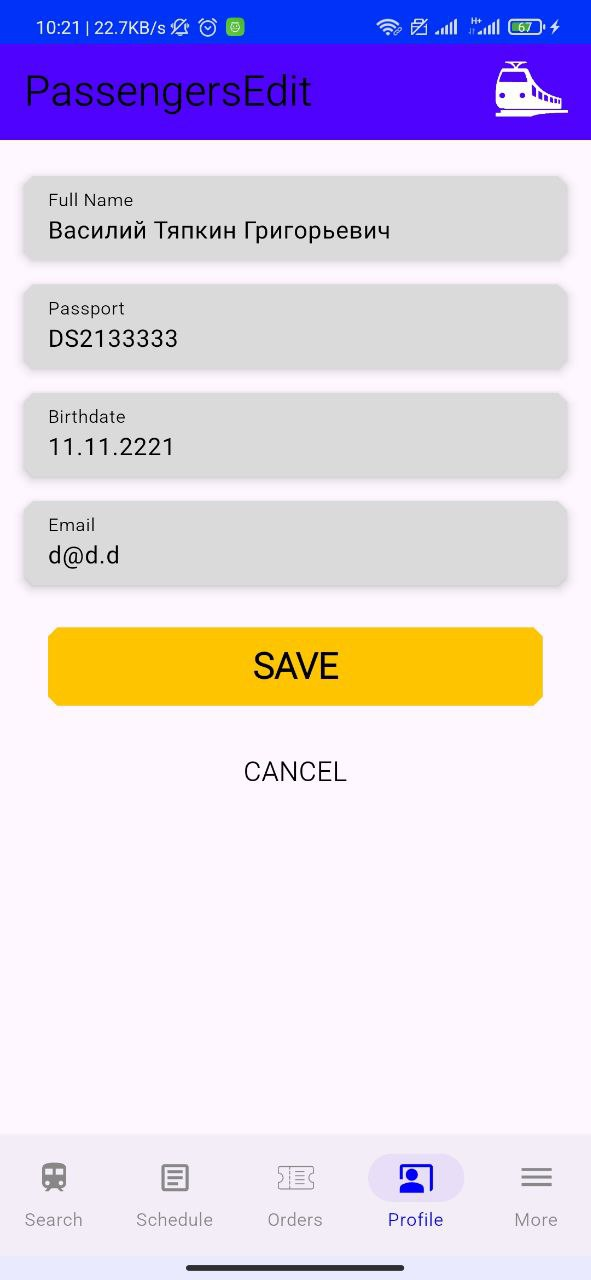
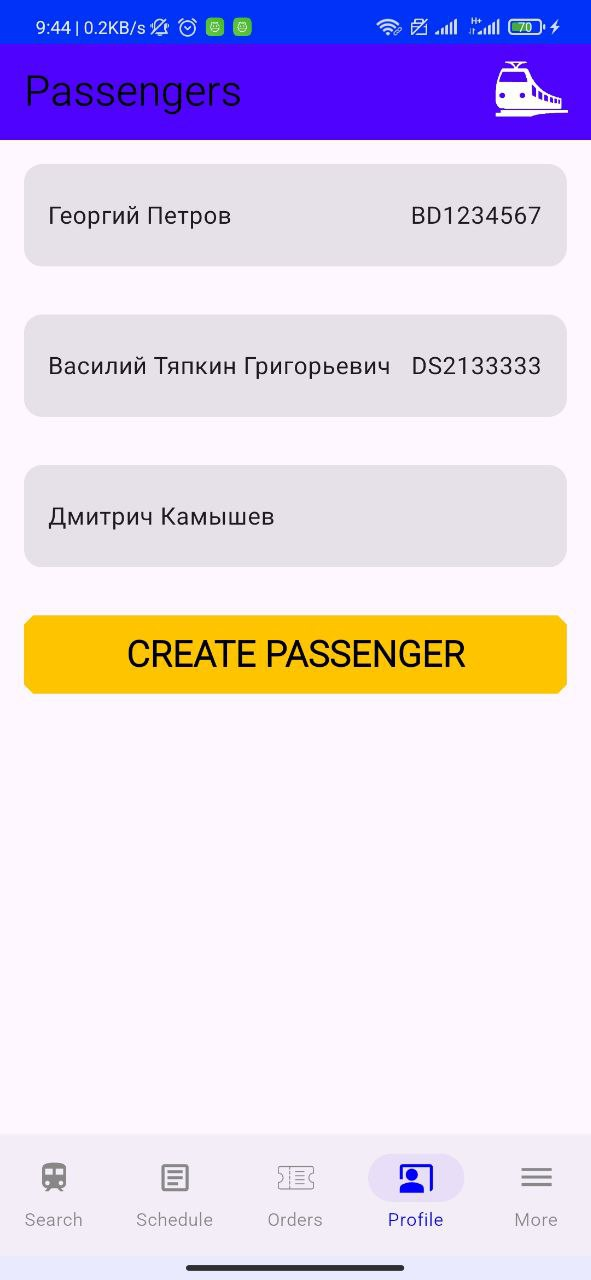


Рисунок 5.3 – Страницы просмотра списка пассажиров и редактирование существующего пассажира

Кроме этого, пользователь может посмотреть расписание отправлений и прибытий поездов. Страница расписания отображена на рисунке 5.4.

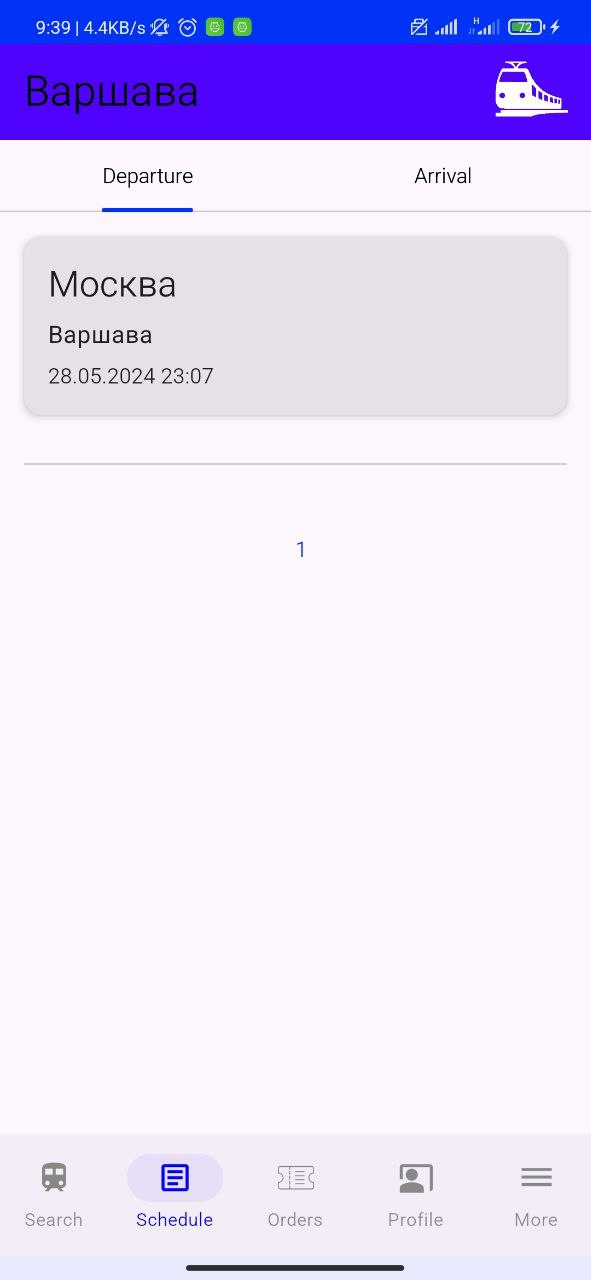


Рисунок 5.4 – Страница расписания поездов

Также пользователь может посмотреть информацию о приложении. Страница «О приложении» показана на рисунке 5.5.

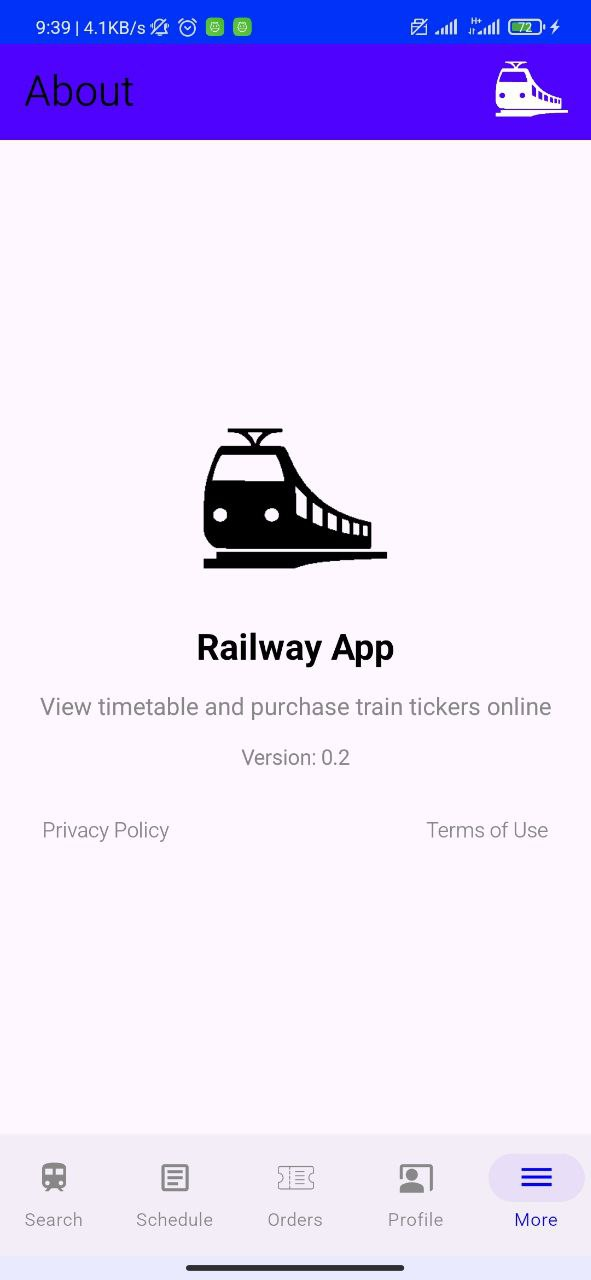


Рисунок 5.5 – Страница «О приложении»

Важной функцей приложения является поиск билетов. Осуществить поиск можно на странице «Поиск». Для этого нужно выбрать станции отправления и прибытия, а также промежуток нужных дат. После этого произойдет перенаправление на страницу результатов поиска, соответсвующих введенным данным. Страницы поиска и найденных маршрутов отображена на рисунке 5.6.

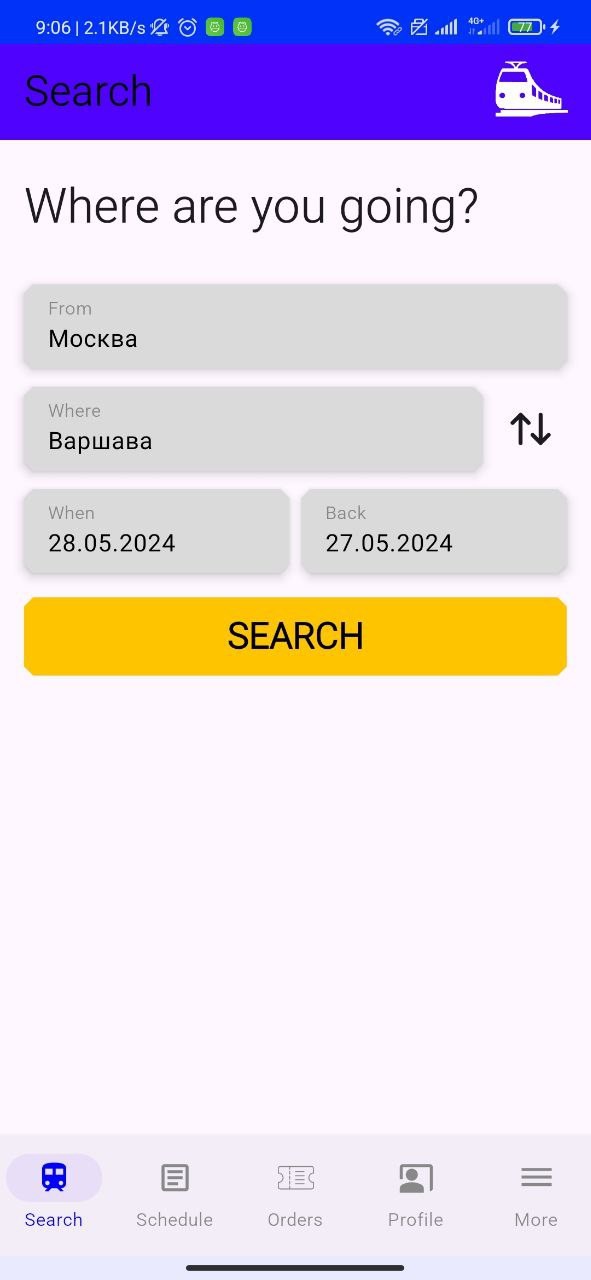
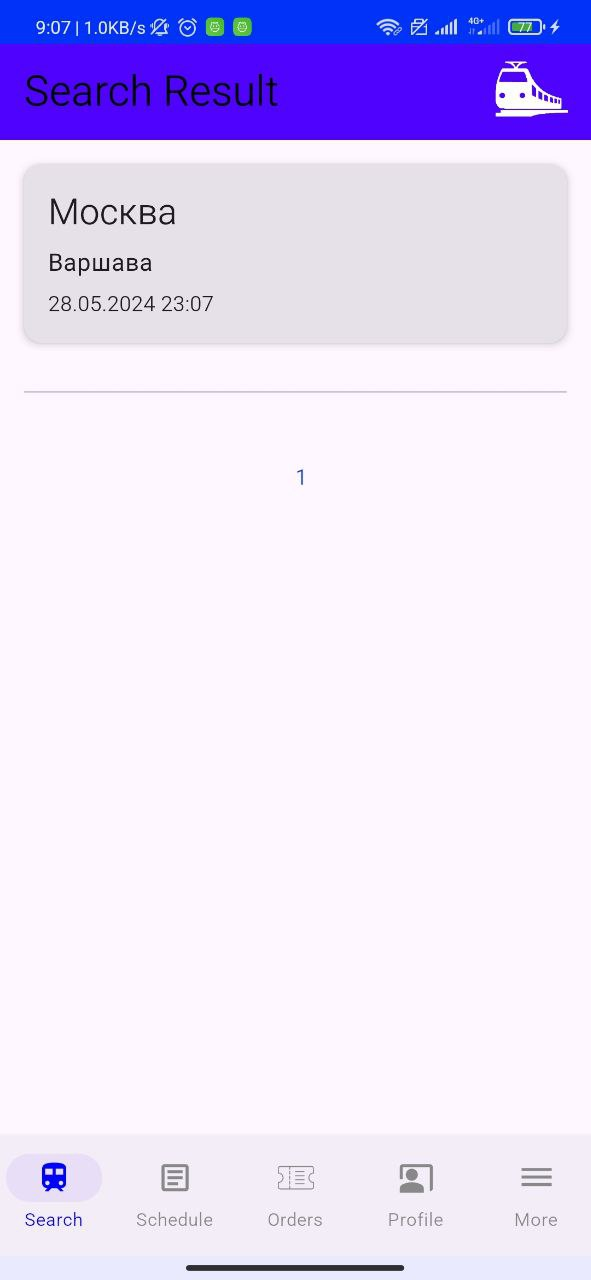
 

Рисунок 5.6 – Страницы поиска и найденных маршуртов

При клике на найденном маршруте открывается страница просмотра свободных мест и бронирования билета. Страница продемонстрирована на рисунке 5.7.

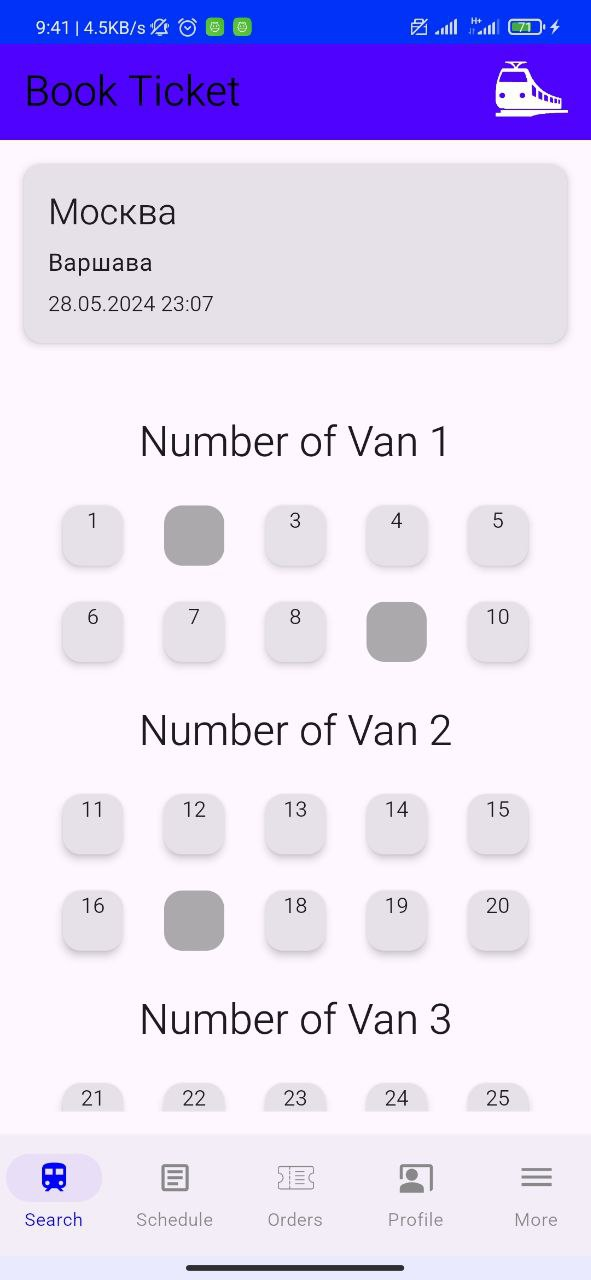


Рисунок 5.7 – Страница просмотра свободных мест и бронирования билетов

После бронирования билета его, вместе с другими забронированными билетами, можно увидеть на странице «Заказы». Если пользователь не вошел в аккаунт, то будет отображено информационное сообщение. Страница показана на рисунке 5.8.

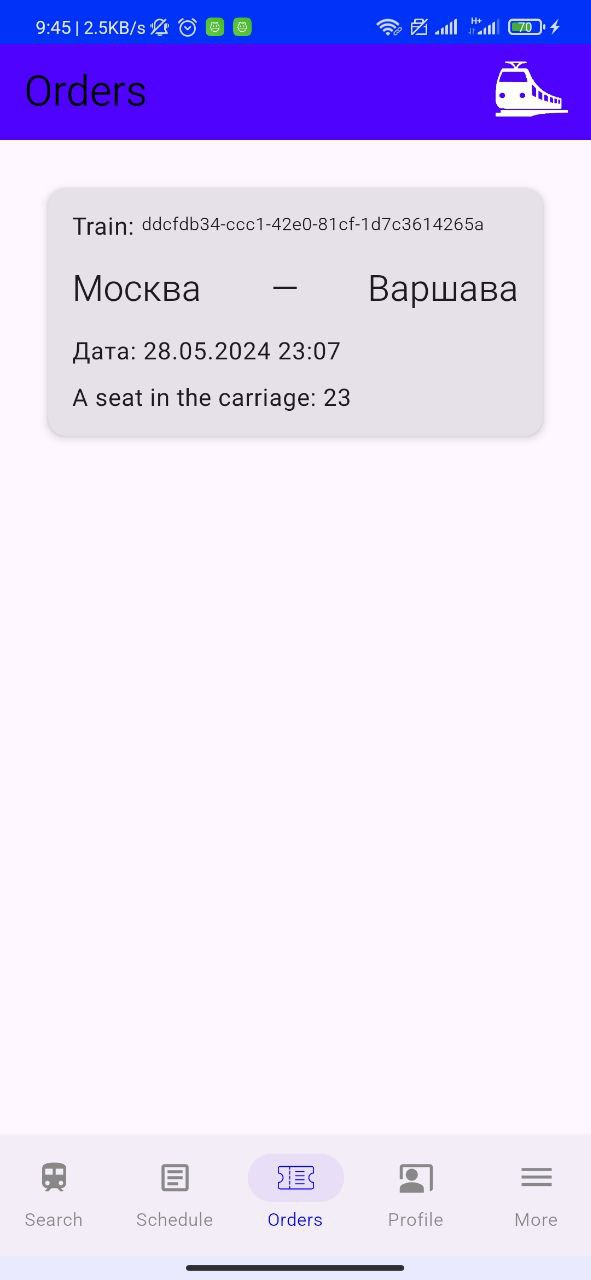


Рисунок 5.8 – Страница «Заказы»

При клике по любому билету открывается детальная информация о билете, включающая QR-код, для сканирования сотрудником железножорожного вокзала и последующей печати билета. Страница детальной информации о билете продемонстрирована на рисунке 5.9.

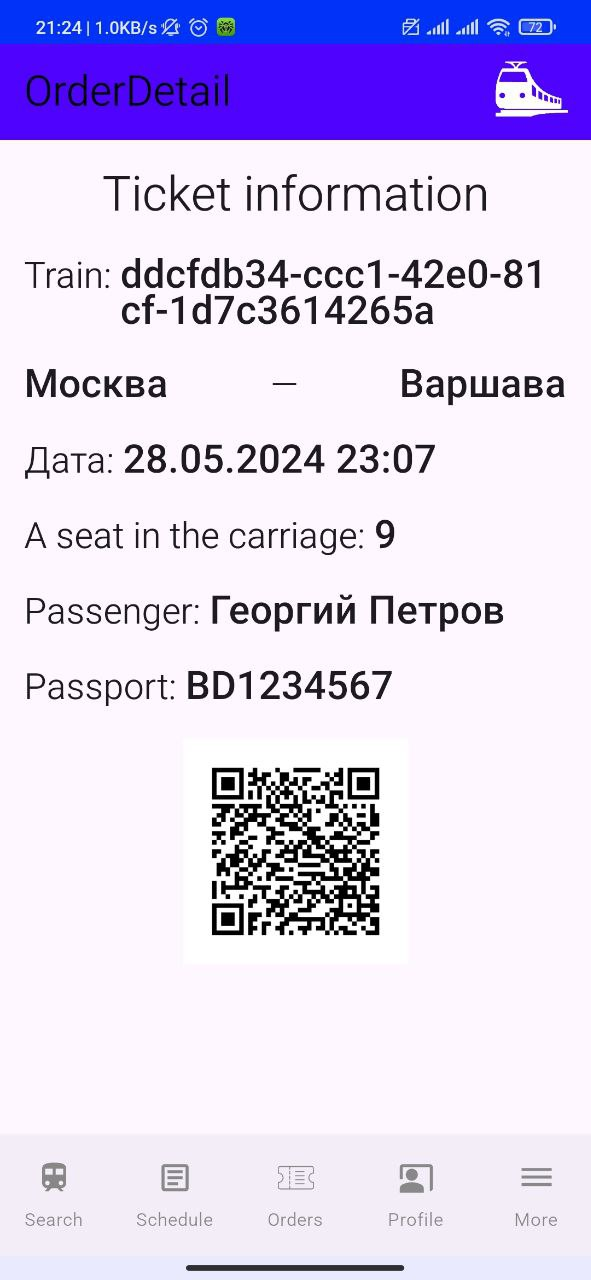


Рисунок 5.9 – Страница «Заказы»

Следующими будут рассмотрены возможности менеджера.

5.1.2 Сторона менеджера

Возможности менеджера такие же, как и возможности обычного пользователя, но к ним добавляются функции добавления, изменения и удаления брони билетов, пассажиров. Для этих операций была специально разработана страница «Админ панель». Она отображена на рисунке 5.10.

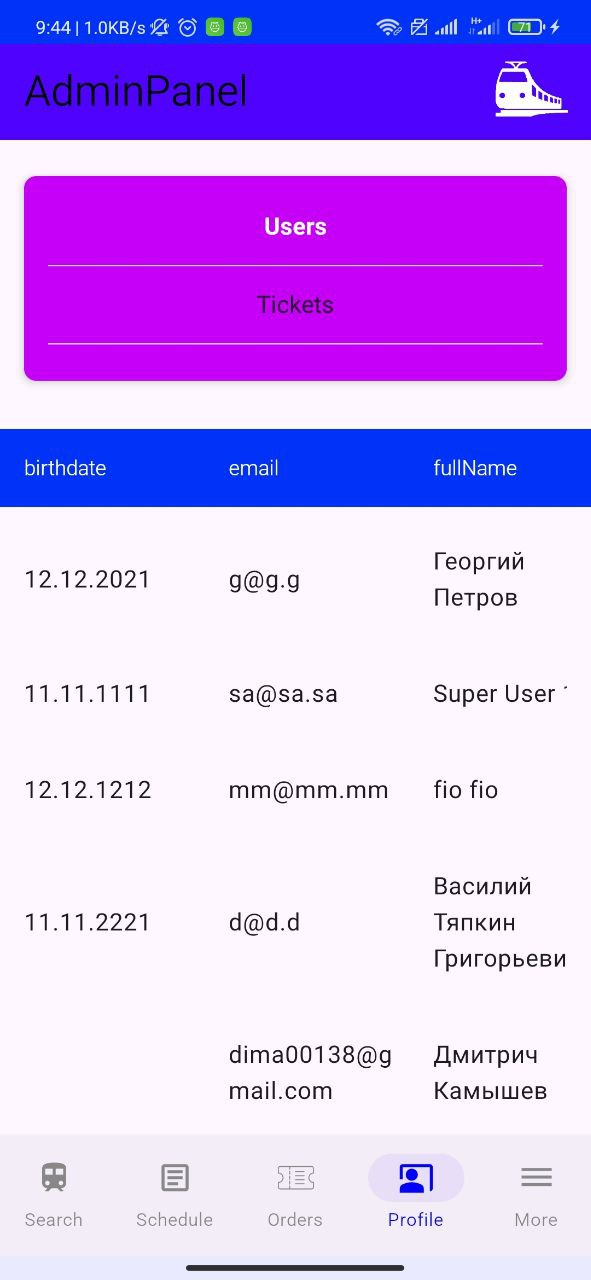


Рисунок 5.10 – Страница «Админ панель»

Если пользователь имеет роль менеджера, то в профиле появляется кнопка, с помощью которой можно попасть на страницу с админ панелью.

5.1.3 Сторона администратора

У администратора свобода добавления, редактирования и удаления для всей информации из базы данных. Страница с админ панелью, для администратора, показана на рисунке 5.11.

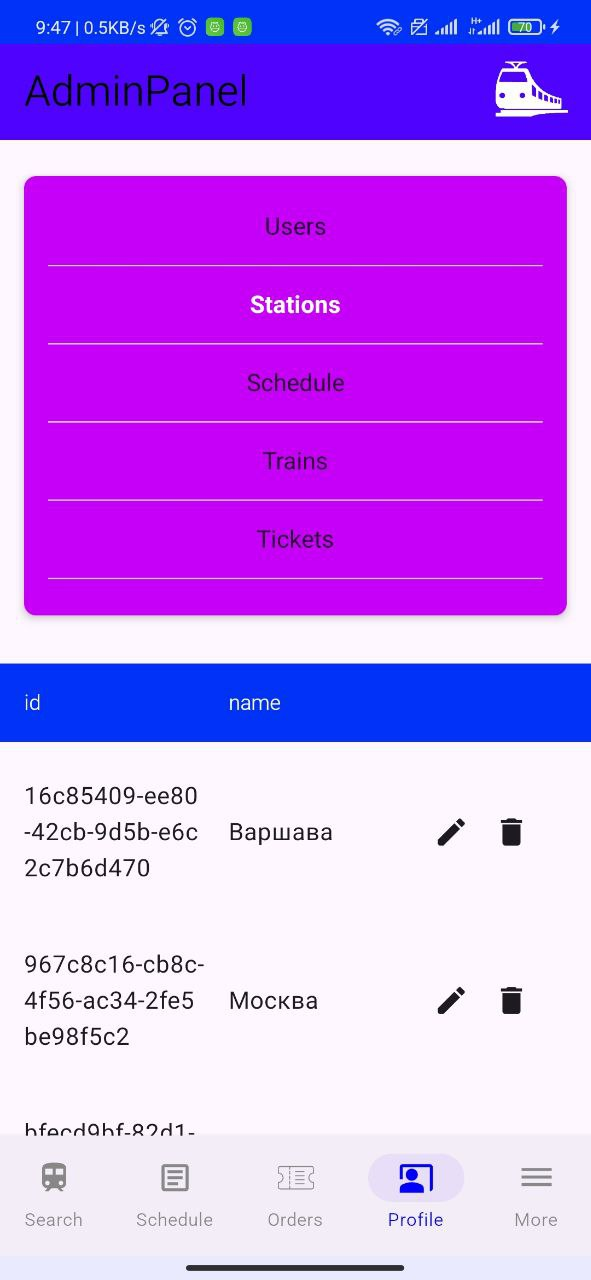


Рисунок 5.11 – Страница «Админ панель»

5.2. Вывод

Было составлено четкое и понятное руководство пользователя для работы с приложением. Были рассмотрены базовые функции приложения, а также предоставлены скриншоты.

Заключение

В данной работе было разработано мобильное приложение. Целью работы было создании удобного и интуитивно понятного интерфейса для пользователей, позволяющего получать актуальную информацию о расписании поездов, изменениях маршрутов и других важных событиях на вокзале.

В рамках работы были поставлены следующие задачи: проектирование приложения и базы данных, разработка мобильного приложения, тестирование и написание руководства пользователя.

Была разработана модель базы данных, состоящая из коллекций, связанных между собой. В модели были учтены основные аспекты работы железнодорожного вокзала, такие как поезда, билеты, пассажиры, станции и т.д.

На основе разработанной модели были созданы необходимые модели данных.

Важным этапом работы было тестирование работоспособности приложения. Были проведены различного рода тесты. Результаты тестирования показали полную работоспособность программного средства.

Было разработано руководство пользователя, которое содержит инструкции по использованию приложения железнодорожного вокзала. В нем описаны основные функции и возможности программного средства.

Как итог, работа позволила создать удобное в использовании и гибкое в управлении приложение, которое может быть использовано для управления данными о расписании и пассажирах, а также может быть адаптировано под конкретные потребности пользователей.

Список использованных литературных источников

1. Design Patterns: Elements of Reusable Object-Oriented Software / Джонсон Ральф, Гамма Эрих [и др.]: СПб.: БХВ-Петербург, 2022. 448 с.
2. БЧ. Мой поезд [Электронный ресурс] / Режим доступа:

https://play.google.com/store/apps/details?id=by.rw.client/ – Дата доступа 08.03.2024.

1. Intercity.pl [Электронный ресурс] / Режим доступа:

https://www.intercity.pl/ – Дата доступа 09.03.2024.

1. Жуков А. А. Мониторинг баз данных СПб.: БХВ-Петербург, 2010. 320 с.
2. Development of Railway Station Management Information System Based on Internet of Things / J. Lin [и др.] // IEEE Access. 2019. №1. С. 25-29.
3. Статья Философия Compose [Электронный ресурс] / Справочник Режим доступа:

https://habr.com/ru/articles/768218/ – Дата доступа 10.03.2024.

1. Форум StackOverflow [Электронный ресурс] / Справочник Режим доступа:

https://ru.stackoverflow.com/ – Дата доступа 17.03.2024.

1. Документация по Jetpack Compose [Электронный ресурс] / Справочник Режим доступа:

https://developer.android.com/develop/ui/compose – Дата доступа 24.03.2024.

1. Документация по Firebase [Электронный ресурс] / Справочник Режим доступа:

https://firebase.google.com/docs – Дата доступа 30.03.2024.

1. Документация по компонентам MaterialUI [Электронный ресурс] / Справочник Режим доступа:

https://m3.material.io/components – Дата доступа 07.04.2024.

1. Статья по интегарции Dagger Hilt в Jetpack compose [Электронный ресурс]/ Справочник Режим доступа:

https://developer.android.com/training/dependency-injection/hilt-android – Дата доступа 16.04.2024.

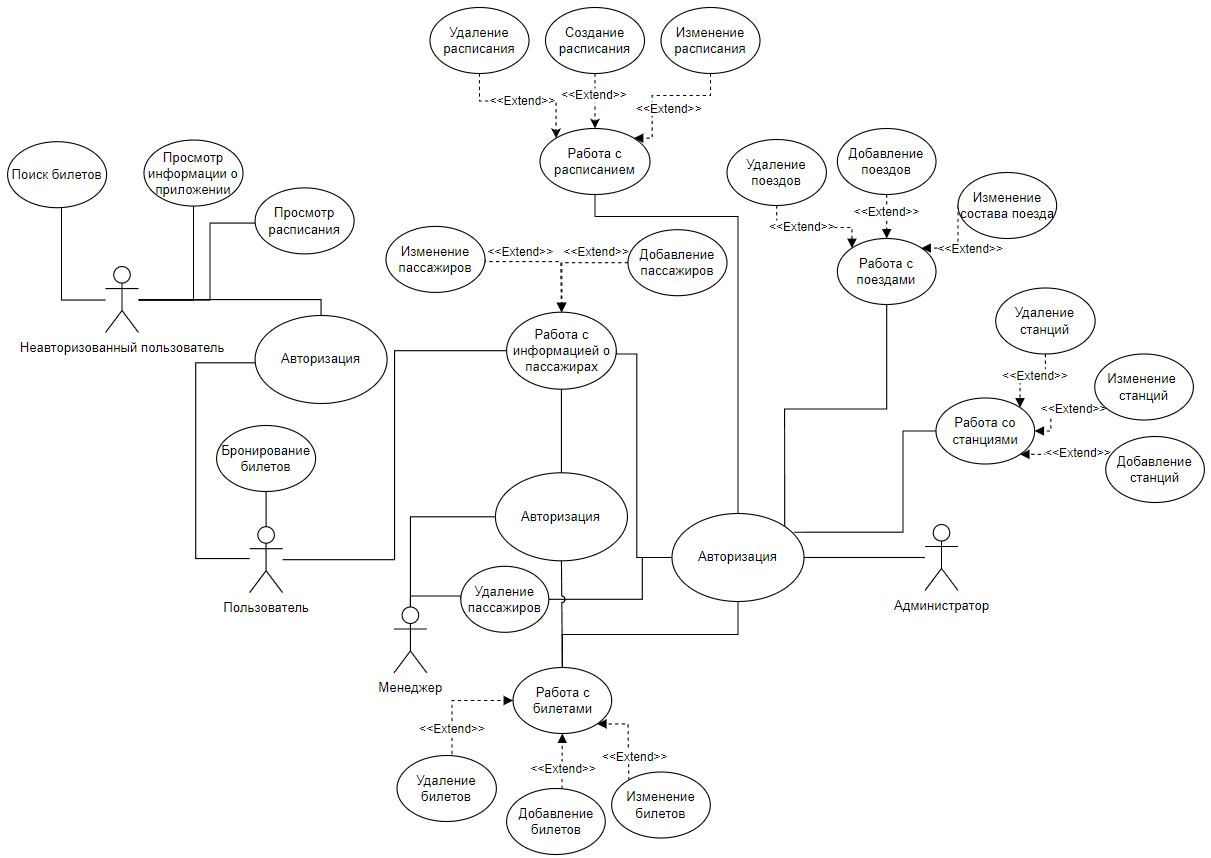
1. Статья StateFlow with Jetpack Compose [Электронный ресурс] / Справочник Режим доступа:

https://farhan-tanvir.medium.com/stateflow-with-jetpack-compose-7d9c9711c286 – Дата доступа 03.05.2024.

Приложение

Приложение А

Диаграмма вариантов использования



Приложение Б

Структура базы данных

